Available online at www.sciencedirect.com
ScienceDirect
Discrete Applied Mathematics 156 (2008) 1846–1855
DISCRETE APPLIED MATHEMATICS
www.elsevier.com/locate/dam

Cost thresholds for dynamic resource location

D. Jacob Wildstrom

Department of Mathematics, University of Louisville, Louisville, KY 40292, USA

Received 20 October 2006; received in revised form 3 July 2007; accepted 2 September 2007
Available online 10 October 2007

Abstract

The traditional dynamic resource location problem attempts to minimize the cost of servicing a number of sequential requests, given foreknowledge of a limited number of requests. One artificial constraint of this problem is the presumption that resource relocation and remote servicing of requests have identical costs. Parameterizing the ratio of relocation cost to service cost leads to two extreme behaviors in terms of dynamic optimizability. The threshold at which a specific graph transitions between these behaviors reveals certain characteristics of the graph's decomposability into cycles.

MSC: 05C85; 90B22; 90B80

Keywords: Dynamic location; Window index; Lookahead; Dynamic optimization

1. Introduction

Practical optimization is usually done with limited knowledge of future events. The class of problems known as *dynamic optimization* or *on-line optimization* seek to model this ignorance. One such problem is the *dynamic location problem*, which models optimal movement of a unique resource on a network to minimize the costs of servicing sequential requests. This problem was originally approached with respect to the development of self-modifying data-structures for search efficiency [4], but has since been explored in the more direct terms of resource management on a graph. One key question in the characterization of a problem in an on-line environment is the extent to which lookahead is useful; in particular, is it possible to devise an optimal set of responses to a request sequence if each response is determined only by knowing the next few requests, and how many future requests are necessary? The required lookahead for an optimal response algorithm on an arbitrary graph has been fully characterized for the dynamic location problem [4,5]. However, the dynamic location problem as traditionally stated assumes that the costs of transporting the resource and of providing service remotely are comparable, which is in many scenarios not indicative of the true costs: for instance, a physical resource on a physical network, such as a printer, may be prohibitively difficult to relocate but inexpensive to access remotely, whereas a high-data-transfer electronic resource, for example a filesystem verifier or database integrity check, may be extremely easy to relocate but use prohibitive amounts of bandwidth to run remotely. In cases where resource-movements and remote-service costs are not equal, we shall see that algorithms for efficient resource location, and the information required to effectively implement those algorithms, vary radically from those used in the traditional solution to this optimization problem. We find that when the dynamic location problem is generalized

E-mail address: djwild01@louisville.edu.

0166-218X/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2007.09.002

in this fashion, the required information for successful optimization is subject to abrupt transitions at certain cost-ratios, and these transition-points are revealing about the nature of the underlying graph.

2. The dynamic location problem generalized

The classical dynamic location problem is that of determining a minimum-cost strategy for relocating a resource on a graph, given an initial location for the resource and a series of requests. That is, given a graph G and input $s_0, r_1, \ldots, r_n \in V(G)$, where s_0 is the initial resource state and $r_1, \ldots, r_n$ are the sequential requests in order, we wish to devise a sequence $s_1, \ldots, s_n \in V(G)$ such that the total cost of moving the resource to each s_i, and servicing r_i in turn, is minimized. The costs associated with movement and remote service are defined to be simply equal to distance in the graph, so that the cost of a request-response pair will be

$$\sum_{i=1}^{n}[d(s_{i-1}, s_i) + d(s_i, r_i)].$$

The problem as formulated thus far, however, neglects the dynamic aspects of the situation. We define a graph G to be *optimizable within window k* if for any initial resource state and request sequence, an optimal response sequence may be determined by selecting s_i as a function of $s_{i-1}, r_i, r_{i+1}, \ldots, r_{i+k-1}$; that is, if the response can be optimized solely based on the current location and the next k requests. The least such k is called the window index, or *windex* of G, written $\mathrm{WX}(G) = k$, and if no such k exists we write $\mathrm{WX}(G) = \infty$.

The graph property $\mathrm{WX}(G)$ has been completely characterized; however, the underlying question assumes that the cost of moving from one node to another and serving that node remotely are identical. As previously mentioned, this is not the case for most situations in which a resource is utilized on a network. A simple modification to the cost-function serves to model this inequity. Knuth [6] suggested the generalized cost function

$$\sum_{i=1}^{n}[c_m d(s_{i-1}, s_i) + c_r d(s_i, r_i)],$$

where c_m and c_r are non-negative constants representing, respectively, the movement costs and remote-service costs associated with this resource. From an optimization standpoint, we may scale our cost function arbitrarily, so scaling by c_r we get the cost function

$$\sum_{i=1}^{n}[\alpha d(s_{i-1}, s_i) + d(s_i, r_i)],$$

where α is the ratio between c_m and c_r, this will correspond to all cases except when $c_r = 0$. The $c_r = 0$ case is demonstrably uninteresting from an algorithmic perspective: if the cost of remote service is zero, then cost can be minimized by simply letting $s_i = s_0$ for all i, which will incur no movement costs. For all other values of c_r, we shall condense c_m and c_r into their ratio α for simplicity. We may calculate minimum-cost response sequences to requests under this new cost, and those optimal response sequences we shall call α-optimal; replacing optimality with α-optimality in the traditional definition of the windex yields a graph parameter we shall call the α-*windex*. We denote the α-windex of a graph G by $\mathrm{WX}^{\alpha}(G)$. Note that the traditional windex is simply the 1-windex under this nomenclature.

In practice, the α-windex for a specific α is rarely illuminating, as for values of $\alpha \neq 1$ it only takes on a small range of values. An inversion of the question of the windex is considerably more revealing: to wit, we want to ask not what the windex is for particular α, but for which α the windex is even finite. We thus define a parameter to determine this critical α-value:

Definition 1. The *windex algorithmic threshold*, denoted $\mathrm{AT}(G)$, is equal to

$$\sup\{\alpha \in [0, 1) : \mathrm{WX}^{\alpha}(G) < \infty\}.$$

Our limitation on the range of α is justified by the exceptional behavior of the α-windex for $\alpha = 1$, which, while the best-understood of the parameterized windex functions, does not actually conform to the behaviors exhibited with all

other values of α. We shall see in the following sections that this parameter determines several graph properties, both familiar and novel.

Theorem 2. *If G is a median graph, then* $\mathrm{AT}(G) = 1$.

Theorem 3. *If G is a non-median graph with finite 1-windex, then* $\mathrm{AT}(G) = \frac{2}{3}$.

Theorem 4. *If G retracts onto either C_{2k} for $k > 2$ or $K_{2,3}$, then* $\mathrm{AT}(G) = \frac{1}{2}$.

Theorem 5. *If G retracts onto C_{2k+1}, then* $\mathrm{AT}(G) \leqslant (k+1)/(2k+1)$.

3. General-use tools

Several properties of the windex generalize to the α-windex, and some other new tools may be used to characterize graphs of particular α-windex. The following two properties are generalizations of tools used in [4].

Definition 6. For graphs G and H, $G\square H$ is a graph such that $V(G\square H) = V(G) \times V(H)$ and $\{(u, v), (u', v')\} \in E(G\square H)$ iff either $u = u'$ and $\{v, v'\} \in E(H)$ or $\{u, u'\} \in E(G)$ and $v = v'$.

Proposition 7. $\mathrm{WX}^\alpha(G\square H) = \max\{\mathrm{WX}^\alpha(G), \mathrm{WX}^\alpha(H)\}$. *Consequentially,* $\mathrm{AT}(G\square K) = \min\{\mathrm{AT}(G), \mathrm{AT}(H)\}$.

Proof. By construction, $d_{G\square H}((u, v), (u', v')) = d_G(u, u') + d_H(v, v')$, so rearranging the terms in the cost, it is clear that the cost of a response $(s_{11}, s_{12}), (s_{21}, s_{22}), \ldots, (s_{n1}, s_{n2})$ to a chosen initial state and request sequence $(s_{01}, s_{02}), (r_{11}, r_{12}), (r_{21}, r_{22}), \ldots, (r_{n1}, r_{n2})$ is simply the sum of the costs of the response $s_{1i}, s_{2i}, \ldots, s_{ni}$ to $s_{0i}, r_{1i}, r_{2i}, \ldots, r_{ni}$ for $i = 1, 2$, so minimizing the cost of a response in $G\square H$ is identical to minimizing the cost of a response in each of G and H.Thus, if response sequences for G and H can be calculated with lookahead k, so can a response sequence in $G\square H$, and conversely. □

Definition 8. A subgraph H of G is a *weak retract* if there is a mapping f of $V(G)$ to $V(H)$ such that for all $u, v \in V(G)$, $d_H(f(u), f(v)) \leqslant d_G(u, v)$, and f restricted to $V(H)$ is the identity.

Proposition 9. *If H is a weak retract of G, then* $\mathrm{WX}^\alpha(H) \leqslant \mathrm{WX}^\alpha(G)$. *Consequentially,* $\mathrm{AT}(H) \geqslant \mathrm{AT}(G)$.

Proof. Given a request sequence $s_0, r_1, r_2, \ldots, r_n \in V(H)$, let $s_1, \ldots, s_n \in V(G)$ be an α-optimal response in G, with cost c. By the definition of a weak retract, the cost of $f(s_1), \ldots, f(s_n)$ as a response to the same request sequence in H is at most c. However, the α-optimal cost of responding to the given sequence in G must be at most the α-optimal cost in H, so the α-optimal cost of that request sequence in H is c and $f(s_1), \ldots, f(s_n)$ is thus an α-optimal response in H; thus, if $s_1, \ldots, s_n$ is determined by lookahead k, we may determine $f(s_1), \ldots, f(s_n)$ with lookahead k as well. □

The above propositions frequently provide lower bounds for $\mathrm{WX}^\alpha(G)$, but to determine upper bounds we will investigate specific α-optimal response sequences. The following tools will help determine the α-optimal responses to specific request sequences.

Remark 10. Let S be the set of all α-optimal s_1 responses to the request configuration $s_0, r_1, r_2, \ldots, r_{n-1}, r_n$ on the graph G, and let S' be the set of all α-optimal s_1 responses to the request configuration $s_0, r_1, r_2, \ldots, r_{n-1}, r'_n$ (that is, a request configuration identical to the first except in the nth request. If $S \cap S' = \emptyset$, then $\mathrm{WX}^\alpha(G) \geqslant n$.

We will use this property by identifying request sequences of arbitrary length differing only in their last term, and to identify the α-optimal response sequences, we shall eliminate the distinctly non-optimal sequences by investigating the cost of their prefixes. It is frequently useful to notate request strings and response sequences as words $s_0 r_1 r_2 \ldots r_n$ and $s_1 s_2 \ldots s_n$ on the alphabet composed of the vertices of the graph.

Proposition 11 (*Prefix Reduction*). *Let $s_0r_1 \dots r_n$ be a request sequence, and let $s_1 \dots s_m s_{m+1} \dots s_n$ and $s'_1 \dots s'_m s_{m+1} \dots s_n$ be response sequences. If the α-costs of $s_1 \dots s_m$ and $s'_1 \dots s'_m$ as a response to $s_0r_1 \dots r_m$ are c and c', respectively, and $c > c' + \alpha d(s_m, s'_m)$, then $s_1 \dots s_m s_{m+1} \dots s_n$ is non-α-optimal.*

Proof. The α-cost of the response sequence $s_1 \dots s_m s_{m+1} \dots s_n$ is

$$c + \alpha d(s_m, s_{m+1}) + d(s_{m+1}, r_{m+1}) + \sum_{i=m+2}^{n} [\alpha d(s_{i-1}, s_i) + d(s_i, r_i)]$$

and the α-cost of $s'_1 \dots s'_m s_{m+1} \dots s_n$ is

$$c' + \alpha d(s'_m, s_{m+1}) + d(s_{m+1}, r_{m+1}) + \sum_{i=m+2}^{n} [\alpha d(s_{i-1}, s_i) + d(s_i, r_i)]$$

so the difference of these costs is $c - c' + \alpha d(s_m, s_{m+1}) - \alpha d(s'_m, s_{m+1})$. The former sequence is non-α-optimal if its α-cost exceeds the α-cost of the latter; that is, if the difference $c - c' + \alpha d(s_m, s_{m+1}) - \alpha d(s'_m, s_{m+1})$ exceeds zero. By the triangle inequality, $\alpha d(s_m, s_{m+1}) - \alpha d(s'_m, s_{m+1}) \geqslant -\alpha d(s_m, s'_m)$, so the above non-α-optimality condition is implied by the inequality $c > c' + \alpha d(s_m, s'_m)$.

Note that for the specific choice of $s_m = s'_m$, the above demonstrates that, in order for $s_1 \dots s_n$ to be an α-optimal response, $s_1 \dots s_m$ must be the minimum-α-cost response to $r_1 \dots r_m$ which has the ending state s_m. Thus, in considering potential α-optimal responses, we may determine potential prefixes by finding the minimum-α-cost response prefixes with specific ending states. □

4. Special cases and trivia

The trivial one-vertex graph has an α-windex of zero for all α, since only one choice of response is possible. Its algorithmic threshold is thus 1. Henceforth in our exploration, we shall only consider connected graphs G with at least two vertices.

Small values of α necessitate finite windex. This will be a useful result, as it establishes that every graph has an algorithmic threshold.

Proposition 12. *The α-windex of any graph is* 1 *for* $\alpha \leqslant \frac{1}{2}$.

Proof. Given a request sequence $s_0r_1r_2 \cdots r_n$ with the response sequence $s_1s_2 \dots s_n$, the cost of servicing these requests is $\sum_{i=1}^{n} \alpha d(s_{i-1}, s_i) + d(s_i, r_i)$. For $1 \leqslant i \leqslant n$, the assumption $\alpha \leqslant \frac{1}{2}$ and the triangle inequality give

$$\begin{aligned}
&\alpha d(s_{i-1}, s_i) + d(s_i, r_i) + \alpha d(s_i, s_{i+1}) \\
&\quad \geqslant \alpha(d(s_{i-1}, s_i) + d(s_i, r_i)) + \alpha(d(r_i, s_i) + d(s_i, s_{i+1})) \\
&\quad \geqslant \alpha d(s_{i-1}, r_i) + d(r_i, r_i) + \alpha d(r_i, s_{i+1}).
\end{aligned}$$

So we see that replacing s_i with r_i in our response sequence does not increase cost; repeating this n times, we see that the response sequence $r_1r_2 \dots r_n$ is no higher in cost than our original response-sequence. Since the original response sequence was arbitrary, it follows that letting $s_i = r_i$ minimizes cost, which can be done without lookahead.

Since $\mathrm{WX}^{1/2}(G) = 1$ for all G, we are assured that the set $\{\alpha \in [\frac{1}{2}, 1) : \mathrm{WX}^{\alpha}(G) < \infty\}$ is non-empty, so that the $\mathrm{AT}(G)$ exists for all G.

The omission of $\alpha = 1$ from the algorithmic-threshold exploration is justified by the complete characterization of that case in [5], but the $\alpha > 1$ case ought to be mentioned briefly, as it in fact describes many of the applications of this model: relocating resources is, in many cases, more expensive than accessing them remotely. □

Proposition 13. *The α-windex of any connected graph on* 2 *or more vertices is infinite for finite* $\alpha > 1$.

Proof. Since K_2 is a weak retract of any non-trivial connected graph, it suffices by Proposition 9 to show that $\mathrm{WX}^{\alpha}(K_2) = \infty$. Let the vertices of K_2 be u and v.

Let $m = \lceil \alpha \rceil$, and consider the request sequences $uv^m(uv)^n u$ and $uv^m(uv)^n v$ for arbitrary n. For $\sigma = uv^m(uv)^n u$, the initial position $s_0 = u$. The minimum-α-cost responses to uv^m with ending states u and v can be determined exhaustively to be u^m and v^m, respectively, with respective costs m and α. By prefix reduction, we know that the α-optimal sequences for the given request sequences begin with one of these two. If we extend this request prefix further to $uv^m uv$, we find that the minimum-α-cost responses for each ending state are u^{m+2} and v^{m+2} with respective costs $m+1$ and $\alpha+1$. Since adding uv to the request string has only lengthened the response patterns and has not affected the relative cost of the two potential α-optimal responses, we may inductively append as many occurrences of uv as desired, so the potential α-optimal response prefixes associated with the request prefix $uv^m(uv)^n$ are u^{m+2n} and v^{m+2n} with respective costs $m+n$ and $\alpha+n$. We may then try each extension possibility of these two to find that the unique α-optimal responses to $uv^m(uv)^n u$ and $uv^m(uv)^n v$ are, respectively, u^{m+2n+1} and v^{m+2n+1}; since these responses differ in the first element, but the requests differ only at the $(m+2n)$th request, a lookahead of at least $m+2n+1$ is necessary to minimize cost. Since n is arbitrary, it thus follows that the windex of K_2 is infinite. □

The above high-relocation-cost case, while not discussed in detail here, is of some significance particularly in the realm of page-migration problems, which, despite the impossibility of perfect service with imperfect information, address the problem of minimizing deviation from optimality. Approximation to within a constant factor of the omniscient optimum has been addressed with both deterministic [3,2] and randomized [8] algorithms. From a threshold-motivated perspective, however, the infinitude of windex when α exceeds 1 serves merely to justify the maximum value of 1 for the algorithmic threshold: regardless of the graph, numbers greater than 1 ought to exceed the algorithmic threshold.

5. Proof of Theorem 2

One of the major classifications integral in establishing windex in [4] and the jumping-off point for the complete characterizations in [5] is the median graph or unique Steiner point property. We shall see that such graphs have finite α-windex for all $\alpha < 1$.

Definition 14. A graph G has the *unique Steiner point property* if for every $u, v, w \in V(G)$, there is a unique $x \in G$ minimizing $d(u,x) + d(v,x) + d(w,x)$.

Definition 15. A graph G is a *median graph* if for every $u, v, w \in V(G)$, there is a unique $x \in G$ such that x lies on a shortest path between each two of $\{u, v, w\}$. Such an x is called the *median point* of u, v, and w.

It is shown in [4] that these two definitions are equivalent, and in [1] that these conditions are equivalent to the property that G is a retract of Q_n for some n. We may thus characterize this large class by α-windex as follows:

Proposition 16. *For* $\frac{1}{2} < \alpha \leqslant 1$, *it follows that* $\mathrm{WX}^\alpha(K_2) = 2$.

Proof. Let the vertices of K_2 be u and v. Clearly $\mathrm{WX}^\alpha(K_2) \geqslant 2$, since exhaustive search reveals that the unique α-optimal response to uvv is vv, whereas the unique α-optimal response to uvu is uu, so a lookahead of at least 2 is necessary. To demonstrate that lookahead 2 is sufficient, we shall use prefix-reduction to determine potential optimal responses given s_k, r_k, and r_{k+1}. Since k is arbitrary, we may simplify by letting $k = 0$, so that we are considering the first three letters of an arbitrary request sequence. w.l.o.g. we may assume $s_0 = u$, so we need only analyze the four possible prefixes uuu, uuv, uvu, and uvv. For the first two, it is clearly optimal to let $r_1 = u$, since the cost of doing so is zero. For the request prefix uvu, the α-optimal response prefixes associated with the two different ending states u and v are uu with cost 1 and vv with cost $1+\alpha$, respectively. We know that uu is at least as low-cost a response as vv, regardless of suffix, since even if it is in a non-ideal to have an ending state of u instead of v, the cost so incurred is at most α. Lastly, for the request prefix uvv, the potential α-optimal response prefixes are uu (with cost 2) and vv (with cost α); by prefix reduction, only vv can ever actually be α-optimal, since the difference in cost exceeds α. Since we can always determine an appropriate r_1 for an α-optimal response string based solely on s_0, r_1, and r_2: thus, $\mathrm{WX}^\alpha(K_2) \leqslant 2$. □

Corollary 17. *For* $\frac{1}{2} < \alpha < 1$, *all non-trivial median graphs have* α*-windex* 2.

Proof. Since $Q_n = Q_{n-1}\Box K_2$, it follows inductively, using Proposition 7, that $\mathrm{WX}^\alpha(Q_n) = 2$ for all n. By Proposition 9, it follows that any retract of Q_n has α-windex at most 2, so, based on the characterizations of [1], every median graph has α-windex at most 2. Since K_2 is a retract of every non-trivial connected graph, it follows that every non-trivial graph also has α-windex of at least 2. □

6. Proof of Theorem 3

Since graphs of finite 1-windex are retracts of products of complete graphs [5], an exploration of the algorithmic properties of graphs of finite 1-windex begins with the algorithmic properties of complete graphs.

Proposition 18. *The α-windex of K_n is 2 for $\frac{1}{2} < \alpha \leqslant \frac{2}{3}$.*

Proof. In the following proof, u, v, and w will describe distinct vertices of K_n (we may consider only the arguments involving request sequences with u and v to show that this works for K_2); x will refer to any vertex not previously mentioned in the request sequence.

That the windex is at least 2 is easy to see: consider the request sequences uvv and uvu. The minimal-cost responses to these are, respectively, uniquely determined to be vv and uu; since they differ in the first response, a lookahead of 2 requests is necessary to determine the first response.

Showing that the windex is at most two requires breaking a response string into two components: we consider both short-term and long-term optimization. Thus, in response to a request, we can consider both the long-term (i.e. which state we want to be in at the end of 2 responses) and the short term (i.e. how to achieve that state at minimum cost). Note that being in an undesirable state would have at most a cost of α, since the option to move to the desired state in response to the third request is always an option.

Without loss of generality, we may assume that $r_0 = u$. If $r_1 = u$, the optimal response is obviously u, since this incurs no cost and leaves all long-term strategies viable. We shall thus consider responses to situations where $r_1 \neq u$; w.l.o.g. we may say that $r_1 = v$.

We have three possible cases to consider: $r_2 = u$, $r_2 = v$, and $r_2 = w$ (which represents any case where r_2 is neither u nor v). For each of these cases we shall craft the short-term optimizations corresponding to each desired state after two responses.

In the case where the request sequence starts with uvu, our first two responses shall be uu, with cost 1, if we want to end in the state u; vv, with cost $1 + \alpha$, if we want to end in the state v; and vx, with cost $1 + 2\alpha$, if we want to end in the state x. Note that both of the latter costs exceed the first by at least α, so no matter what state we wish to be in by the third request, uu is an optimal response prefix to a request starting with uvu.

In the case where the request sequence starts with uvv, our first two responses shall be uu, with cost 2, if we want to end in the state u; vv, with cost α, if we want to end in the state v; and vx, with cost $1 + 2\alpha$, if we want to end in the state x. As before, one of these response strings (vv) has a cost which is less than the others by at least α, so regardless of what state we wish in the third and later requests, this response prefix is optimal.

Lastly, in the case where the request sequence starts with uvw, our first two responses shall be uu, with cost 2, if we want to end in the state u; vv, with cost $1 + \alpha$, if we want to end in the state v; vw, with cost 2α, if we want to end in the state w; and vx, with cost $1 + 2\alpha$, if we want to end in the state x. We may eliminate as before those strings which cost 3α or more, since vw is preferable even in the long term; this leaves vv and vw as viable responses, so some response prefixed with v is optimal.

Thus, regardless of r_3 and later requests, we may determine s_1 optimally from s_0, r_1, and r_2. Performing this task iteratively, we may determine optimal s_ℓ from $s_{\ell-1}$, r_ℓ, and $r_{\ell+1}$, so K_n has a windex of at most 2. □

From the above, we can use known properties of the 1-windex to demonstrate that a relationship between the 1-windex and α-windex.

Corollary 19. *If $\mathrm{WX}^1(G) < \infty$ and G is non-trivial, then $\mathrm{WX}^\alpha(G) = 2$ for $\frac{1}{2} < \alpha \leqslant \frac{2}{3}$.*

Proof. Since G is non-trivial, it follows that $\mathrm{WX}^\alpha(G) \geqslant 2$. It was shown in [5] that $\mathrm{WX}^1(G) = k$ iff G is a retract of $K_k\Box K_k\Box K_k\Box\cdots\Box K_k$. Since $\mathrm{WX}^\alpha(K_k) = 2$ for any k, it follows from Propositions 7 and 9 that $\mathrm{WX}^\alpha(G) \leqslant 2$. □

It thus follows that for $\mathrm{WX}^1(G) < \infty$, $\mathrm{AT}(G) \geqslant \frac{2}{3}$. That this is an equality follows from the result, shown in Section 8, that $\mathrm{AT}(K_3) = \frac{2}{3}$, so since K_3 is a retract of G, $\mathrm{AT}(G) \leqslant \frac{2}{3}$ as well.

7. Proof of Theorem 4

There are two varieties of graph which have infinite α-windex for even small values of α. If G retracts onto either of these, it will inherit their algorithmic intractability.

Proposition 20. *For $k > 2$, if $\frac{1}{2} < \alpha < 1$, then $\mathrm{WX}^\alpha(C_{2k}) = \infty$, and thus $\mathrm{AT}(C_{2k}) = \frac{1}{2}$.*

Proof. Let u, a, b and c be vertices in C_{2k} such that $d(u, a) = 1$, $d(u, c) = d(u, b) = k - 1$, and $d(a, c) = k - 2$, as shown in Fig. 1. Then we shall consider the request sequences $ua(bc)^n b$ and $ua(bc)^n c$. The first request is at a in both cases, so our first response can plausibly be u, a, or one vertex beyond a. No other response can possibly be optimum, since movement further than u from a clearly increases cost for the first request, but provides no benefit that could not be realized by performing the desired relocation on the second response as well. The vertex one beyond a is, for the same reason, at best as good as u, so we may limit our consideration to the possibilities of responses beginning with u or a.

If $s_1 = a$, then the remainder of the response sequence is a response to $a(bc)^n b$ or $a(bc)^n c$. Since a, b, and c all lie on one half of C_{2k}, we may safely ignore the other half of C_{2k} as not relevant to optimization (since relocation by way of that half is at best as good as relocation through the half containing a, b, and c. Thus, we may complete our optimizations on a P_{k+1} rather than a C_k. The response algorithm on a path is comparatively simple and identical for all $\alpha \in (\frac{1}{2}, 1]$: in this situation, all further responses would be c, until perhaps the last. Thus, subject to $s_1 = a$, the α-cost-optimizing responses to $ua(bc)^n b$ and $ua(bc)^n c$ are $ac^{2n}b$ and ac^{2n+1}, respectively, which have respective α-costs of $(k + 1)\alpha + 2n$ and $(k - 1)\alpha + 2n$.

On the other hand, if $s_1 = u$, then the remainder of the response sequence is a response to $u(bc)^n b$ or $u(bc)^n c$. It is clearly optimal for the first response to be b, c, or the point in between, so as to minimize distance to all the remaining requests at b and c. Henceforth we can restrict our consideration to the P_3 containing these three points and easily determine the α-cost-minimizing responses subject to $s_1 = u$ to be ub^{2n+1} and $ub^{2n-1}c^2$, with respective α-costs of $(k - 1)\alpha + 2n + 1$ and $(k + 1)\alpha + 2n - 1$.

Thus, in response to the sequence $ua(bc)^n b$, we may choose either $s_1 = a$ or $s_1 \neq a$ (represented by the case $s_1 = u$ above), leading to minimum α-costs of $(k + 1)\alpha + 2n$ and $(k - 1)\alpha + 2n + 1$, respectively. The latter is smaller, so every possible α-optimal response to $ua(bc)^n b$ has $s_1 \neq a$.

Similarly, in response to $ua(bc)^n c$, we may choose either $s_1 = a$ or $s_1 \neq a$, with respective minimum α-costs of $(k - 1)\alpha + 2n$ and $(k + 1)\alpha + 2n - 1$. The former is smaller, so every possible α-optimal response to $ua(bc)^n c$ has $s_1 = a$. Since these two sequences differing only in their $(2n + 2)$th request require different choices of s_1 in α-optimal responses, it follows that $\mathrm{WX}^\alpha(C_{2k}) \geqslant 2n + 2$. Since n is arbitrary, it follows that $\mathrm{WX}^\alpha(C_{2k}) = \infty$. □

Proposition 21. $\mathrm{WX}^\alpha(K_{2,3}) = \infty$ *for* $\frac{1}{2} < \alpha < 1$, *and thus* $\mathrm{AT}(K_{2,3}) = \frac{1}{2}$.

Proof. Let the vertices in the 2-vertex part be x and y; let those in the 3-vertex part be u, v, and w. Consider the request sequences $u(vw)^n x$ and $u(vw)^n y$. Response cost to requests alternating between v and w is optimized when the resource state is on the shortest path between v and w, so to optimize the cost for arbitrarily high n, we would wish to choose s_1 to be v, w, x, or y, then remain in this position for all states until $s_{2n+1} = r_{2n+1}$. Thus, our potential α-optimal responses to $u(vw)^n x$ are $v^{2n}x$, $w^{2n}x$, x^{2n+1}, and $y^{2n}x$, with respective costs of $3\alpha + 2n$, $3\alpha + 2n$, $\alpha + 2n$, and $3\alpha + 2n$;

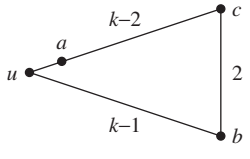

Fig. 1. Request locations on C_{2k} for Proposition 20.

thus x^{2n+1} is the unique α-optimal response. Similarly we can establish that y^{2n+1} is the unique α-optimal response to $u(vw)^n y$, so since a difference in the $(2n+1)$th request induces a difference in the first response, $\mathrm{WX}^\alpha(K_{2,3}) \geqslant 2n+1$, and since n is arbitrary, $\mathrm{WX}^\alpha(K_{2,3}) = \infty$. □

Theorem 4 follows naturally from Proposition 9 and the fact that the algorithmic threshold can be no smaller than $\frac{1}{2}$.

8. Proof of Theorem 5

Proposition 22. *For $k \geqslant 1$, if $(k+1)/(2k+1) < \alpha < 1$, then $\mathrm{WX}^\alpha(C_{2k+1}) = \infty$. Thus $\mathrm{AT}(C_{2k+1}) \leqslant (k+1)/(2k+1)$.*

Proof. Note that $(k+1)/(2k+1) > \frac{1}{2}$, so we will henceforth assume that $\alpha \in (\frac{1}{2}, 1)$. Let u, v, and w be vertices in C_{2k+1} such that $d(u, v) = 1$ and $d(v, w) = d(w, u) = k$, as shown in Fig. 2. Then we shall consider the request sequences $uvw(uw)^n u$ and $uvw(uw)^n w$. We shall start by noting that at no time is any point midway along of any of the paths between u, v, and w an α-optimal response to these request sequences: if the long-term benefit of traversing a path exceeds the cost of movement along the path, then traversal of the entire path is warranted; that is, if movement away from one request and towards another results in improved cost, improvement is maximized by moving all the way along the edge. Thus we may consider responses which contain only the resource states u, v, and w.

If $s_1 = v$, then the α-optimal choice of s_2 must be w: if it were u, then the response prefix vu could be changed to uu reducing α-cost by $2\alpha - 1$, and if it were v, then s_3 would be clearly optimally u, and once again the prefix vvu could be modified to uuu, reducing α-cost by $2\alpha - 1$. Henceforth, since s_2 and $r_3, \ldots, r_n$ are all on the unique shortest path between u and w, we may simply find the α-optimal response on P_k: there may be multiple such response sequences, but one such is that which remains unchanging until the end, i.e. $s_2 = s_3 = s_4 = \cdots = s_{n-1}$ and $s_n = r_n$. Thus, representative possible α-optimal responses to $uvw(uw)^n u$ and $uvw(uw)^n w$ subject to the condition that $s_1 = v$ are $vw^{2n+1}u$ and vw^{2n+2}, respectively, with respective α-costs $kn + (2k+1)\alpha$ and $kn + (k+1)\alpha$. Note that these sequences may not be α-optimal overall, but are the least α-cost sequences subject to the constraint $s_1 = v$.

If $s_1 = u$, then the since s_1 and $r_2, \ldots, r_n$ are all on the unique shortest path between u and w, we may again restrict to P_k in order to optimize the remaining responses, so representative possible α-optimal responses to $uvw(uw)^n u$ and $uvw(uw)^n w$ subject to the condition that $s_1 = u$ are u^{2n+3} and uw^{2n+2}, respectively, with respective α-costs $1 + k(n+1)$ and $1 + kn + k\alpha$.

We may omit the case $s_1 = w$ from consideration, as $d(w, v) \geqslant d(u, v)$, and thus any case in which $s_1 = w$ has an α-cost of at least the same as $s_1 = u$, and we may thus consider these cases in tandem with the $s_1 = u$ cases, that is, as an $s_1 \neq v$ case.

Thus, in response to the sequence $uvw(uw)^n u$, we may choose either $s_1 \neq v$ or $s_1 = v$, leading to minimum α-costs of $1 + k(n+1)$ and $kn + (2k+1)\alpha$ respectively. Since $\alpha > (k+1)/(2k+1)$, it follows that $1 + k(n+1) < kn + (2k+1)\alpha$, so every possible α-optimal response to $uvw(uw)^n u$ has $s_1 \neq v$.

Similarly, in response to $uvw(uw)^n w$, we may choose either $s_1 \neq v$ or $s_1 = v$, with respective minimum α-costs of $1 + kn + k\alpha$ and $kn + (k+1)\alpha$. Since $\alpha < 1$, $1 + kn + k\alpha > kn + (k+1)\alpha$, so every possible α-optimal response to $uvw(uw)^n w$ has $s_1 = v$. Since these two sequences differing only in their $(2n+3)$th request require different choices of s_1 in α-optimal responses, it follows that $\mathrm{WX}^\alpha(C_{2k+1}) \geqslant 2n+3$. Since n is arbitrary, it follows that $\mathrm{WX}^\alpha(C_{2k+1}) = \infty$. □

This bound on the algorithmic threshold is in fact sharp:

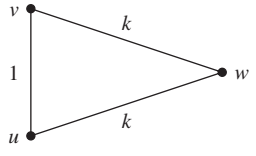

Fig. 2. Request locations on C_{2k+1} for Proposition 22.

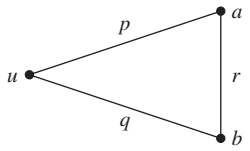

Fig. 3. Request locations on C_{2k+1} for Proposition 23.

Proposition 23. *For* $k \geqslant 1$, *if* $\frac{1}{2} < \alpha < (k+1)/(2k+1)$, *then* $\mathrm{WX}^{\alpha}(C_{2k+1}) = 2$. *Thus* $\mathrm{AT}(C_{2k+1}) \geqslant (k+1)/(2k+1)$.

Proof. Let us consider a two-request prefix uab on C_{2k+1}, where u and a are connected by a path of length p, u and b by a path of length q, and a and b connected by a path of length r, so that $p+q+r = 2k+1$ as shown in Fig. 3. To allow the greatest generality possible in this sequence, u, a, and b may not be all distinct, so any of p, q, or r could be zero. First of all, let us note that if $p \geqslant q+r$, no optimal response traverses the path ua, so we can, from an algorithmic perspective, consider this situation identical to one in which we remove an edge from the path ua, which will reduce to the path P_{2k+1}. Since P_{2k+1} is a median graph, we can find optimal responses therein with a lookahead of 2, so these cases will not be algorithmically complicated enough to require a windex of greater than 2. Likewise we may dispense with any case where $q \geqslant p+r$ or $r \geqslant p+q$.

Therefore, we need only consider responses to this prefix in which p, q, and r conform to the triangle inequality. As in the previous proof, the only responses we need consider are those involving the vertices mentioned in the request sequence, since if movement of a request along one of the paths reduces cost, the cost reduction will continue until it reaches the end of the path, and thus the minimal-cost response will necessarily involve an endpoint of the path. This consideration yields nine possible response prefixes, since there are three vertices and two requests.

Several of the responses can be rejected out of hand. The response ua is higher-cost than aa; au and bu are higher cost than uu; and ba is higher-cost that aa; we are left with only five potential response-prefixes: uu, ub, aa, ab, and bb. Since we are dealing with a request/response pair as a prefix to a possibly longer sequence, the end-state of the response is of interest. The end-states u and a are uniquely associated with uu and aa, respectively. The end-state b may be associated with any of 3 different responses of different costs.

The costs of the responses ub, ab, and bb are $p+\alpha q$, $\alpha p+\alpha r$, and $r+\alpha q$ respectively. Note that $(1-\alpha)/\alpha = (p+q+r-1)/(p+q+r+1) \geqslant (2p-2)/2p$ since $p+q+r-1 \geqslant 2p$ and $(x-1)/(x+1)$ is an increasing function. Then

$$\frac{1-\alpha}{\alpha}p \geqslant \frac{2p-2}{2p}p = p-1 \geqslant r-q$$

so, redistributing terms, $p+\alpha q \geqslant \alpha p+\alpha r$; likewise we can show $r+\alpha q \geqslant \alpha p+\alpha r$. Thus, the least α-cost response with an end-state of b is ab.

We thus have three viable response-prefixes depending on the desired end-state: uu with cost $p+q$, aa with cost $\alpha p+r$, and ab with cost $\alpha p+\alpha r$. Since the distance between u and b is q, prefix reduction ensures that the prefix ab with an additional expenditure of αq can be considered to have the same end-state as uu. Then note that since $\alpha < (p+q+r+1)/(2p+2q+2r)$,

$$\alpha p+\alpha q+\alpha r \leqslant \frac{p+q+r+1}{2} \leqslant p+q$$

so the response prefix uu is no better than ab. Thus, given this request prefix, regardless of what follows, the response prefix aa or ab will always be appropriate. Since we can determine the first response solely from the first two requests, we may thus assert that the α-windex of C_{2k+1} is 2. □

9. Characterization by odd cycles

Since odd cycles (and thus products of odd cycles) have highly characteristic algorithmic thresholds, the algorithmic threshold of a given graph may illuminate its retractability onto an expression as a retraction of odd cycles. For example, the following proposition follows simply from a threshold argument:

Proposition 24. *No even cycle of more than* 4 *vertices, or odd cycles of more than* $2\ell + 1$ *vertices can be expressed as a retraction of a product of odd cycles of lengths at most* $2\ell + 1$.

Proof. Let $G = C_{2k_1+1}\Box C_{2k_2+1}\Box\cdots\Box C_{2k_r+1}$; by Theorem 5 and Proposition 7, $\mathrm{AT}(G) = \min((k_1 + 1)/(2k_1 + 1), (k_2+1)/(2k_2+1), \ldots, (k_r+1)/(2k_r+1)) = (\min\{k_i\}+1)/(2\ \min\{k_i\}+1) = (\ell+1)/(2\ell+1)$. So by Proposition 9, if H is a retract of G, then $\mathrm{AT}(H) \geqslant (\ell + 1)/(2\ell + 1)$. Since $\mathrm{AT}(C_n) < (\ell + 1)/(2\ell + 1)$ for even $n > 4$ or odd $n > 2\ell + 1$, H cannot be a cycle of such length.

It is established in [7] that every graph can be retracted to a minimal cycle, and the possible retractions of a graph place an upper bound on its algorithmic threshold; we may thus establish that graphs of algorithmic threshold larger than $\frac{1}{2}$ necessarily lack minimal cycles of even order; furthermore, if a graph's algorithmic threshold exceeds $\frac{2}{3}$, then it is triangle-free, and graphs with significantly larger values of the algorithmic threshold have even greater restrictions on the orders of cycles present therein. □

Thus, the algorithmic threshold of a given graph reveals significant information about its expression as a retraction, in particular of Q_n, and its own retractability onto various cycles.

References

[1] H.-J. Bandelt, Retracts of hypercubes, J. Graph Theory 8 (4) (1984) 501–510.
[2] Y. Bartal, M. Charikar, P. Indyk, On page migration and other relaxed task systems, Theoret. Comput. Sci. 268 (1) (2001) 43–66, on-line algorithms (Udine, 1998).
[3] M. Chrobak, L.L. Larmore, N. Reingold, J. Westbrook, Page migration algorithms using work functions, J. Algorithms 24 (1) (1997) 124–157.
[4] F.R.K. Chung, R.L. Graham, M.E. Saks, Dynamic search in graphs, Discrete Algorithms and Complexity (Kyoto, 1986), Perspectives in Computing, vol. 15, Academic Press, Boston, MA, 1987, pp. 351–387.
[5] F.R.K. Chung, R.L. Graham, M.E. Saks, A dynamic location problem for graphs, Combinatorica 9 (2) (1989) 111–131.
[6] D.E. Knuth, Personal communication (2005).
[7] R. Nowakowski, I. Rival, Fixed-edge theorem for graphs with loops, J. Graph Theory 3 (4) (1979) 339–350.
[8] J. Westbrook, Randomized algorithms for multiprocessor page migration, in: On-line algorithms (New Brunswick, NJ, 1991), DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 7, American Mathematical Society, Providence, RI, 1992, pp. 135–149.